**CYBER ATTACK DETECTION USING DIFFERENT RESOURCES**

*A Mini Project Report submitted to*

*JNTU Hyderabad in partial fulfillment*

*of the requirements for the award of the degree*

**BACHELOR OF TECHNOLOGY**

In

**ELECTRONICS AND COMMUNICATION ENGINEERING**

***Submitted by***

**MARPU PRATHYUSHA 20RG1A0494**

**DUSSA SAHITHI 21RG5A0405**

**KONGANDLA MANJULA 21RG5A0407**

**KONGARA KAVYASRI 21RG5A0408**

*Under the Guidance of*

***Mrs. K . SUREKHA***

*M.Tech (VLSISD).*

*Assistant Professor*



***DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING***

**MALLA REDDY COLLEGE OF ENGINEERING FOR WOMEN**

*(Approved by AICTE New Delhi and Affiliated to JNTUH)*

*(An ISO 9001: 2015 Certified Institution)*

*(B.Tech Programs(CSE,ECE) Accredited by NBA)*

*Maisammaguda, Medchal (M), Hyderabad-500100, T.S*

*SEPTEMBER 2023*

***DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING***

**MALLA REDDY COLLEGE OF ENGINEERING FOR WOMEN**

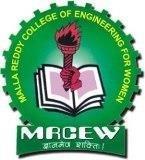
*(Approved by AICTE New Delhi and Affiliated to JNTUH)*

*(An ISO 9001: 2015 Certified Institution)*

*(B.Tech Programs(CSE,ECE) Accredited by NBA)*

*Maisammaguda, Medchal (M), Hyderabad-500100, T. S.*

*SEPTEMBER 2023*



***CERTIFICATE***

This is to certify that the Mini project entitled **“CYBER ATTACK DETECTION USING DIFFERENT RESOURCES”** has been submitted by **Marpu Prathyusha (20RG1A0494), Dussa Sahithi (21RG5A0405), Kongandla Manjula (21RG5A0407),Kongara Kavyasri (21RG5A0408)** in partial fulfillment of the requirements for the award of **BACHELOR OF TECHNOLOGY** in **ELECTRONICS & COMMUNICATION ENGINEERING**. This record of bonafide work was carried out by them under my guidance and supervision. ***The result embodied in this mini-project report has not been submitted to any other University or Institute for the award of any degree.***

**Mrs. K. Surekha.** **Dr. Selvamani Indrajith**

*Assistant Professor Head of the Department*

*Project Guide*

***External Examiner***

**ACKNOWLEDGEMENT**

The Mini Project work carried out by our team in the Department of Electronics and Communication Engineering, Malla Reddy College of Engineering for Women, Hyderabad. ***This work is original and has not been submitted in part or full for any degree or diploma of any other university.***

We wish to acknowledge our sincere thanks to our project guide

***Mrs. K. Surekha***, Assistant Professor of Electronics & Communication Engineering Department for formulating the problem, analysis, guidance and her continuous supervision during the project work.

We acknowledge our sincere thanks to ***Dr. Vaka Murali Mohan***, Director, MRCEW. ***Dr. Kanaka Durga Returi*,** Principal and Professor of Computer Science & Engineering Department. ***Dr.******Selvamani Indrajith***, Professor and Head of the Electronics & Communication Engineering Department, for their kind cooperation in making this Mini Project work a success.

We extend our gratitude to ***Sri. Ch. Malla Reddy***, Founder-Chairman and ***Sri. Ch. Mahender Reddy****,* Secretary, MRCEW and ***Sri. P. Praveen Reddy*** Director, MRGI for their kind cooperation in providing the infrastructure for successful completion of our Mini project work.

We convey our special thanks to the Entire Teaching faculty and non- teaching staff members of the Electronics & Communication Engineering Department for their support in making this project work a success.

**MARPU PRATHYUSHA (20RG1A0494)**

**DUSSA SAHITHI (21RG5A0405)**

**KONGANDLA MANJULA (21RG5A0407)**

**KONGARA KAVYASRI (21RG5A0408)**

INDEX

|  |  |
| --- | --- |
| **Chapter** | **Page No** |
| **ABSTRACT** | **iii** |
| **LIST OF FIGURES** | **iv** |
| **LIST OF TABLES** | **V** |
| **1.INTRODUCTION** | **1** |
| 1.1 Back ground Introduction | 1 |
| 1.2 Goals | 1 |
| 1.3 Objectives | 2 |
| **2.LITERATURE SURVEY** | **8** |
| **3.EXISTING SYSTEM** | **11** |
| **4.PROPOSED SYSTEM** | **12** |
| **5.MODULE DESCRIPTION**  5.1 system architecture  5.1 UML diagram | **14**  14  14 |
| **6.SYSTEM REQUIREMENTS** | **18** |
| 6.1 Hardware Requirements | 18 |
| 6.2 Software Requirements | 18 |
| **7.RESULTS AND DISCUSSIONS** | **19** |
| **8.CONCLUSION & FUTURE SCOPE** | **28** |
| **9. APPENDIX** | **29** |
| **10. BIBLIOGRAPHY** | **42** |

ABSTRACTTop of Form

In the need for an effective automated method to identify virtual threats. To tackle this, we introducing an AI-based approach that relies on artificial neural networks. This technique involves converting large volumes of gathered threat incidents into individual event profiles, which then serve as the foundation for an advanced risk characterization process. We are establish an AI-SIEM framework that integrates event profiling, preprocessing algorithms, and various artificial neural network strategies. This framework not only separates genuine alerts from false positives but also prompts security experts to swiftly respond to virtual hazards. The study's methodology is validated using two benchmark datasets (NSLKDD and CICIDS2017) alongside real-world data. Through evaluations against five common AI techniques (SVM, k-NN, RF, NB, and DT), the paper demonstrates that the proposed methods outperform traditional approaches. Ultimately, the research highlights the viability of using learning-based models for network intrusion detection and underscores that these techniques prove more effective even in practical applications. The central focus is on an AI-SIEM approach for threat detection, and the study's evaluation involves comparing its performance against algorithms like SVM, Decision Tree, Random Forest, k-NN, and Naïve Bayes.

Top of Form

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIG.NO** | **FIGURE NAME** | **PAGE NO** |
| 1.1 | Supervised Learning | 5 |
| 1.2 | Unsupervised Learning | 6 |
| 1.3 | Machine Learning Process | 7 |
| 4.1 | Block Diagram of Proposed Cyberattack Detection | 12 |
| 5.1 | System architecture | 14 |
| 5.2 | use case diagram | 15 |
| 5.3 | sequence diagram | 15 |
| 5.4 | Activity flow diagram | 16 |
| 5.5 | Deployment diagram | 17 |
| 5.6 | Component diagram | 17 |
| 7.1 | Uploading diagram | 19 |
| 7.2 | TF-IDF Algorithm | 20 |
| 7.3 | Generate event vector | 21 |
| 7.4 | SVM Algorithm | 21 |
| 7.5 | KNN Algorithm | 22 |
| 7.6 | Random forest Algorithm | 22 |
| 7.7 | Naïve Bayes | 23 |
| 7.8 | Decision Tree Prediction | 23 |
| 7.9 | Accuracy comparing graph | 24 |
| 7.10 | Precision comparing graph | 24 |
| 7.11 | Re-call comparing graph | 25 |
| 7.12 | Fmeasure comparing graph | 25 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO** | **TABLE NAME** | **PAGE NO** |
| 1.1 | Difference between supervised and unsupervised learning | 5 |

**CHAPTER1: INTRODUCTION**

* 1. **Background Introduction:**

With the evolution of artificial intelligence (AI), particularly in the realm of intrusion detection, there has been significant progress in identifying digital attacks. However, the ever-evolving nature of cyber threats poses challenges in safeguarding IT systems against malicious activities. Two primary approaches for detecting threats are intrusion prediction systems (IPS) and security event analysis. IPS operates within enterprises, generating intrusion alerts based on network patterns. Security event analysis involves scrutinizing unusual alerts to uncover malicious behavior. Yet, advanced attacks remain difficult to detect due to their sophisticated nature.To address this, AI and machine learning techniques have been employed. However, current methods face limitations. They often require labeled data, which can be hard to obtain in large volumes. Many SIEM solutions lack labeled data, hindering the creation of effective models. Also, some models produce numerous false positives, consuming resources during analysis. Attackers can adapt and evade detection, rendering existing models ineffective. Additionally, most methods focus on short-term data analysis, while long-term security event records might hold key insights

**1.2 Goals:**

The main goal is to bolster IT system security using AI techniques. AI excels at spotting advanced cyber threats that traditional methods might miss, automating threat analysis to save time for experts. Challenges arise from evolving attacks, but AI's adaptive nature helps it learn from past incidents and adjust to changing tactics. Reducing false alarms is crucial; AI minimizes incorrect alerts for better resource allocation. Long-term analysis and detailed event profiles enable AI to uncover complex attacks over time, enhancing overall cybersecurity.

**1.3 OBJECTIVES**

**Transforming Security Activities into Event Profiles**: The primary aim of this research is to convert various security activities and incidents into singular event profiles. By doing so, the study aims to streamline the handling of vast amounts of security-related data. This transformation simplifies the analysis process by condensing complex information into easily interpretable profiles, allowing security experts to efficiently navigate and make sense of the data.**Generalizable Event Analysis Method**: The research focuses on developing a method that can analyze security events in a generalizable manner. This approach aims to capture both common and rare patterns from a diverse range of collected data. By identifying recurrent events and understanding their frequency of occurrence, the method seeks to establish a comprehensive understanding of different types of security incidents. This can provide insights into trends, anomalies, and potential threats that might not be immediately apparent using traditional methods.**Utilizing AI Techniques for Event Profiling:** A significant aspect of this research involves harnessing the power of artificial intelligence techniques for event profiling. The goal is to leverage AI algorithms to automatically identify crucial features and attributes within event profiles. By doing so, the method aims to enhance the accuracy of event categorization, allowing for more sophisticated analysis. This approach empowers security experts to delve deeper into the data, uncovering hidden patterns that could be indicative of potential threats.Bottom of Form**What Is Artificial intelligence?**

Artificial Intelligence (AI) refers to the simulation of human intelligence processes by computer systems. It involves creating algorithms and systems that enable computers to perform tasks that typically require human intelligence, such as reasoning, problem-solving, learning from experience, understanding natural language, and recognizing patterns. AI technologies aim to mimic cognitive functions, enabling machines to analyze data, make decisions, and solve problems in ways that simulate human thinking. AI encompasses various subfields, including machine learning, natural language processing, computer vision, robotics, and more. It has applications in a wide range of industries, from healthcare and finance to manufacturing and entertainment, with the goal of automating complex tasks, improving efficiency, and advancing human capabilities.

**How Does AI Work?**

Artificial intelligence (AI) operates through intricate algorithms that enable computers to replicate human-like cognitive functions. Machine learning, a significant component of AI, involves training algorithms on data to recognize patterns and make predictions or decisions. This data-driven approach consists of several steps:Data Collection: Relevant data is gathered, often in large quantities, from various sources.**Data Preprocessing:** Raw data is cleaned, transformed, and organized to ensure its quality and suitability for analysis.**Feature Extraction**: Important features or attributes are identified within the data, serving as inputs for AI models.**Algorithm Selection:** Depending on the task, appropriate algorithms are chosen. For instance, neural networks are effective for complex pattern recognition, while decision trees excel in making decisions based on rules.**Training:** The selected algorithm is trained using labeled data. It adjusts its internal parameters iteratively to minimize errors and improve accuracy.**Validation and Testing:** The trained model is evaluated using unseen data to ensure it can generalize its learning beyond the training dataset.**Fine-Tuning:** If necessary, the model's performance is enhanced through adjustments in algorithms, parameters, or features.**Deployment:** Once the model performs well, it's deployed for practical applications. For example, an AI system can classify emails as spam or not spam.**Inference:** In the deployment phase, the model uses new, unlabeled data to make predictions or decisions based on its training.**Feedback Loop:** Continuous learning allows models to adapt to changing data over time, refining their accuracy and insights.

**What is machine Learning?**

Machine learning serves as a pivotal subset of Artificial Intelligence (AI), where applications learn from data without explicit programming. This entails computers autonomously discovering insights when exposed to fresh information. By utilizing iterative processes and algorithms, machine learning enables computers to adapt and evolve independently. This concept, although rooted in historical examples like the Enigma Machine during World War II, has gained significant traction in recent years. In essence, machine learning encompasses the capacity to self-adjust through successive data iterations, relying on pattern recognition to deliver dependable and informed outcomes.

**How Does Machine Learning Work?**

Machine Learning is, undoubtedly, one of the most exciting subsets of Artificial Intelligence. It completes the task of learning from data with specific inputs to the machine. It’s important to understand what makes Machine Learning work and, thus, how it can be used in the future. The Machine Learning process starts with inputting training data into the selected algorithm.

Training data being known or unknown data to develop the final Machine Learning algorithm. The type of training data input does impact the algorithm, and that concept will be covered further momentarily. New input data is fed into the machine learning algorithm to test whether the algorithm works correctly. The prediction and results are then checked against each other.

**What are the Different Types of Machine Learning?**

Machine Learning is complex, which is why it has been divided into two primary areas, supervised learning and unsupervised learning. Each one has a specific purpose and action, yielding results and utilizing various forms of data. Approximately 70 percent of machine learning is supervised learning, while unsupervised learning accounts for anywhere from 10 to 20 percent. The remainder is taken up by reinforcement learning.

Table -1.1: Difference between Supervised and Unsupervised Learning

|  |  |  |
| --- | --- | --- |
| **Factors** | **Supervised learning** | **Unsupervised learning** |
| Input | Known and Labeled data | Unknown data |
| Complexity | Very complex | Less complex |
| Number of classes | Known | Unknown |
| Accuracy | Accurate and reliable | Moderately Accurate and reliable |

**1. Supervised Learning:**

In supervised learning, we use known or labeled data for the training data. Since the data is known, the learning is, therefore, supervised, i.e., directed into successful execution. The input data goes through the Machine Learning algorithm and is used to train the model. Once the model is trained based on the known data, you can use unknown data into the model and get a new response.

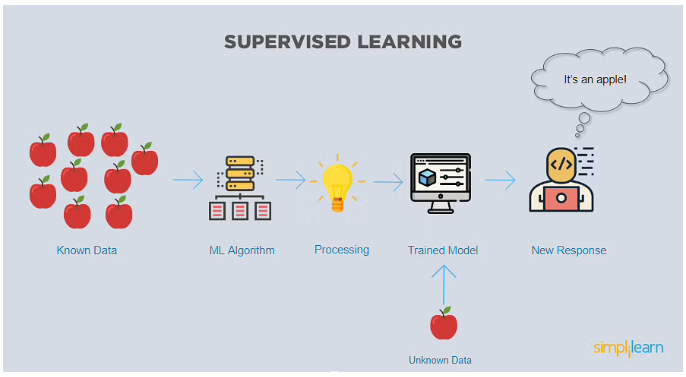


fig 1.1: Supervised Learning

In this case, the model tries to figure out whether the data is an apple or another fruit. Once the model has been trained well, it will identify that the data is an apple and give the desired response.

Here is the list of top algorithms currently being used for supervised learning are:

1. support vector machine

2. Random Forest

3. Linear regression

4. Decision trees

6. K-nearest neighbors

7. Naive Bayes

**2. Unsupervised Learning:**

In unsupervised learning, the training data is unknown and unlabeled - meaning that no one has looked at the data before. Without the aspect of known data, the input cannot be guided to the algorithm, which is where the unsupervised term originates from. This data is fed to the Machine Learning algorithm and is used to train the model. The trained model tries to search for a pattern and give the desired response. In this case, it is often like the algorithm is trying to break code like the Enigma machine but without the human mind directly involved but rather a machine.

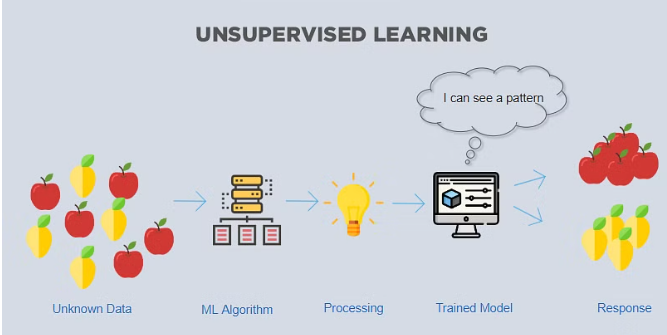


Fig1.2: Unsupervised Learning

**Why is Machine Learning Important?**

To better answer the question “what is machine learning” and understand the uses of Machine Learning, consider some of the applications of Machine Learning: the self-driving Google car, cyber fraud detection, and online recommendation engines from Facebook, Netflix, and Amazon. Machines make all these things possible by filtering useful pieces of information and piecing them together based on patterns to get accurate results.

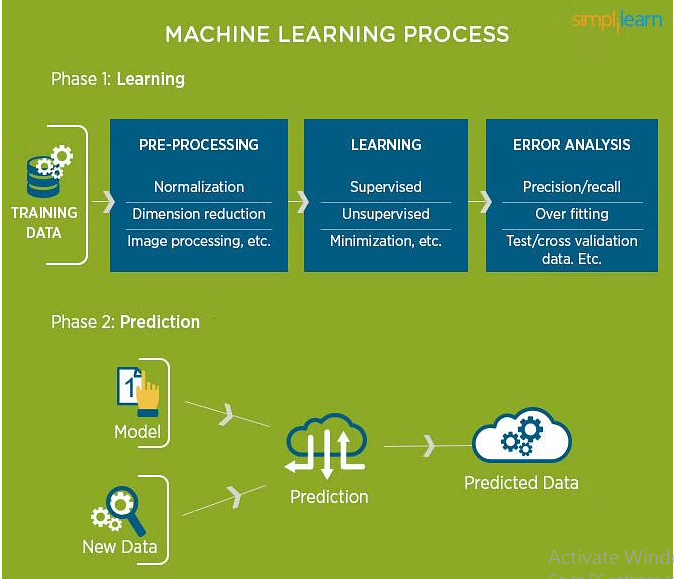


fig1.3: Machine Learning Process

**Main Uses of Machine Learning:**

Typical results from machine learning applications usually include web search results, real-time ads on web pages and mobile devices, email spam filtering, network intrusion detection, and pattern and image recognition.

All these are the by-products of using machine learning to analyze massive volumes of data

**CHAPTER 2: LITERATURE SURVEY**.

**S. Naseer, Y. Saleem, S. Khalid, M. K. Bashir, J. Han, M. M. Iqbal, and K. Han[1]** Interference acknowledgment is significant for local area circumstance care. While a few procedures were proposed to recognize network interference, they can't clearly and fittingly utilize semi-quantitative measurements including handle insights and quantitative records. Therefore, this paper proposes some other personality rendition relying upon an organized non-cyclic diagram (DAG) and a conviction rule base (BRB). In the proposed model, called DAG-BRB, the DAG is used to construct a multi-layered BRB rendition that can hold an essential separation from impact of blends of rule amount in mellow of unlimited assortments of interference. To gather the ideal impediments of the DAG-BRB model, an advanced necessity covariance local area adaption headway framework (CMA-ES) is developed up which could effectively manage the basic issue in the BRB. A relevant assessment changed into applied to check the adequacy of the proposed DAG-BRB. The outcomes showed that differentiated and other region molds, the DAG-BRB model has a superior distinguishing proof rate and can be used in genuine organizations. In this paper, another DAG-BRB adaptation become proposed for network interference character and a crucial CMA-ES estimation turned out to be in like manner created to assemble the limits of this model.

The responsibilities of this paper can be summarized as follows: Directed non-cyclic outlines (DAGs) are applied inside the DAGBRB model, in which some BRB models shape a blend model, and the alluded to assessments of each BRB adaptation are considered as an element of the limits to be updated. The ER rule and the disadvantage CMA-ES counts are used to set up the constraints of each BRB model inside the DAG-BRB. A relevant.

**S. S. Sekharan and K. Kandasamy[2]**Nowadays, IT affiliations produce monster proportions of information. Dealing with those bits of records itself is essential in the IT worldwide. Consequently fusing the log the chiefs system improves security in this manner upgrades information guarantee in an affiliation. Such endeavors require a high profiling gadget that encourages in managing the data and events realities to improve the recognition of insurance. Security Information and Event Management (SIEM) is a technique for assurance examination that sizeable excellent a characterize of insurance in an affiliation. SIEM contraptions gather, study, normalize and relates all insights and harm down realities coming from the outstanding gadget and supply a focused disposition on logs. This paper expresses a consultation of SIEM devices and occasion association vehicles, equipping a diagram of their specific similar assessment, focusing in on most celebrated SIEM units and open inventory rule-basically based dating vehicles and profiles them. SIEM systems have come to be an indispensable piece of organization watchmen to recognize, respond, and criminal of scenes.

**W. Wang, Y. Sheng, and J. Wang[3]**The improvement of an eccentricity based interference disclosure system (IDS) is a urgent assessment heading inside the order of interference notoriety. An IDS learns standard and strange direct through looking at network traffic and can find hard to comprehend and new attacks. Regardless, the presentation of an IDS is significantly challenge to envelop plan, and making arrangements a rundown of capacities that may correctly portray network traffic is as however a persistent investigation issue. Peculiarity essentially based IDSs also have the trouble of a high false alert cost (FAR), which truly restricts their even minded bundles. In this paper, we advocate a one of a kind IDS alluded to as the cutting edge spatial-brief features based interference ID structure (HAST-IDS), which to begin with learns the low-degree spatial features of big business guests using significant convolutional neural offices (CNNs) and in some time learns quickened level common features using long impermanent memory associations.

**B. Zhang, G. Hu, Z. Zhou, Y. Zhang, P. Qiao, and L. Chang[4]** The authors propose a method that leverages a combination of Directed Acyclic Graph (DAG) and Belief Rule Base (BRB) for association-based intrusion detection. The primary idea is to enhance the detection of intrusions in a network by utilizing the structure of a directed acyclic graph to represent relationships among network entities and leveraging a belief rule base to make informed decisions about intrusion events.By integrating DAG and BRB, the method aims to improve the accuracy of intrusion detection by effectively modeling the associations and dependencies between various network elements. The paper likely delves into the technical details of how the directed acyclic graph is constructed, how the belief rule base is utilized for decision-making, and how this integrated approach enhances intrusion detection performance.

**Y. Shen, E. Mariconti, P. Vervier, and Gianluca Stringhini[5]**Tiresias: Predicting Security Events Through Deep Learning" by Y. Shen, E. Mariconti, P. Vervier, and Gianluca Stringhini, presented at ACM CCS 18 in Toronto, Canada, introduces the Tiresias framework. This framework utilizes deep learning to forecast security events, leveraging machine learning models to predict various security incidents. The authors detail the framework's approach, including data sources, deep learning model design, and evaluation. The focus is on enhancing cybersecurity through proactive event prediction using advanced machine learning techniques.

**R. Vinayakumar, Mamoun Alazab, K.P.Soman, P.Poorna[6]** Significantly Learning Approach for Intelligent Intrusion Detection System" suggests a research paper or study that focuses on enhancing intrusion detection systems using a significant learning approach. While the specific details of the paper are not provided in your input, the title implies that the authors have explored a novel method for improving the effectiveness and accuracy of intrusion detection in computer systems. The term "significant learning" often refers to the identification and utilization of key patterns, features, or data points that contribute significantly to the understanding and performance of a system, which in this case, would be an intrusion detection system. This work likely contributes to the field of cybersecurity by proposing innovative techniques to better detect and respond to unauthorized activities within computer network.

**CHAPTER 3: EXISTING SYSTEM**

Existing methods for cyberattack detection include computational intelligence approaches, model-based detection, signature-based detection, and heuristic detection. However, each method has its own limitations. Here are some of the limitations of existing methods for cyberattack detection from the search results:

**Limitations of computational intelligence approaches:** Computational intelligence approaches, such as machine learning, can be limited by the quality and quantity of data used to train the models. In addition, these approaches may not be able to detect new or previously unknown threats that do not fit into the existing pattern.

**Limitations of model-based detection:** Model-based detection methods rely on accurate models of the system being monitored, which can be difficult to obtain and maintain. In addition, these methods may not be able to detect attacks that do not affect the system's physical state.

**Limitations of signature-based detection:** Signature-based detection methods rely on known attack signatures, which can be easily evaded by attackers who modify their attacks. In addition, these methods may not be able to detect new or previously unknown threats.

**Limitations of heuristic detection:** Heuristic detection methods rely on identifying patterns or anomalies in the system's behaviour, which can result in false positives or false negatives. In addition, these methods may not be able to detect attacks that do not exhibit unusual behaviour.

**Limitations of state estimation-based detection**: State estimation-based detection methods rely on accurate models of the system's physical state, which can be difficult to obtain and maintain. In addition, these methods may not be able to detect attacks that do not affect the system's physical state.

**CHAPTER 4: PROPOSED SYSTEM**

The project research centered around intrusion detection and network security, focusing on the application of AI and machine learning techniques. It notes that recent studies have concentrated on using AI for intrusion detection, but often within limited datasets. In contrast, some researchers have employed real-world security events and logs for their analysis, providing more realistic insights. The text introduces a new approach categorized into direct and combination methods for intrusion detection, using a structured non-cyclic graph belief model. This approach incorporates both network-level and host-level analysis, utilizing a Deep Neural Network (DNN) model that is compared to traditional AI methods in terms of effectiveness. The study aims to enhance intrusion detection for actual security scenarios by leveraging AI's capabilities.

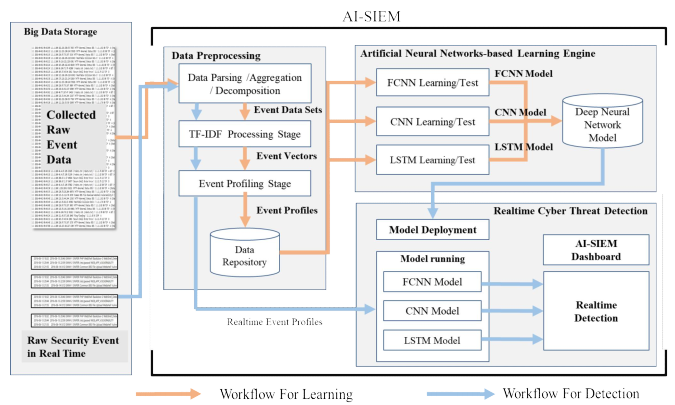


fig 4.1: The Block Diagram of Cyberattack Detection

**AI-SIEM Structure and Purpose:**

The AI-SIEM (Artificial Intelligence Security Information and Event Management) system aims to enhance cybersecurity by automating the analysis of security events. By utilizing advanced machine learning techniques and preprocessing methods, the system can handle a vast array of network events. The primary goal is to identify potential security threats, generate relevant alerts, and facilitate the understanding of virtual risks.

**Data Preprocessing:**

Data preprocessing plays a pivotal role in preparing raw security event data for analysis. The initial stage, referred to as "event profiling," involves transforming raw data into a format suitable for neural networks. This process likely involves extracting key features, normalizing data, and possibly reducing dimensionality. This prepared data serves as input to the subsequent analysis stages.

**GPU Utilization**:

The integration of graphical processing units (GPUs) into the AI-SIEM system demonstrates a commitment to accelerating the analysis process. GPUs are well-suited for tasks that require parallel processing, such as complex calculations and pattern recognition. Their inclusion in the system architecture enhances the system's overall efficiency and speed, enabling faster and more accurate threat analysis.

**AI-Driven Detection Engine:**

The heart of the AI-SIEM system is its detection engine, which employs artificial neural networks to identify patterns and anomalies in security events. This AI-based engine likely

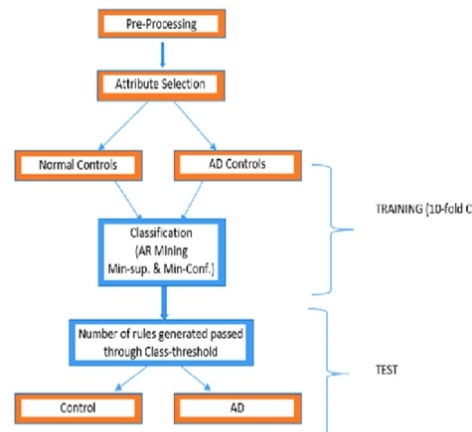
**Data Labelling for Learning:**

Supervised learning, a machine learning approach, relies on labelled data to train models effectively. In the context of the AI-SIEM system, each security event needs to be labelled as either "Normal" or "Threat." However, manual labelling of a large volume of security events is resource-intensive and time-consuming.

**Automated Labelling:**

To address the challenges of manual labelling, the AI-SIEM system employs automated labelling techniques. Instead of relying solely on human analysts to label events, the system likely employs heuristics and algorithms to categorize events based on their characteristics.

**CHAPTER 5: MODULE DESCRIPTION**

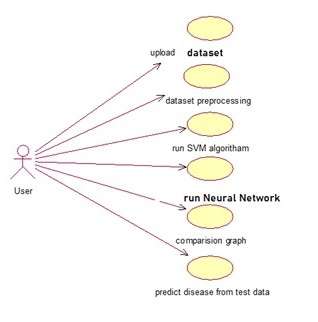
**5.1 SYSTEM ARCHITECTURE:**

5.1 flow diagram

**5.2 UML DIAGARMS :**

**USE CASE DIAGRAM:**

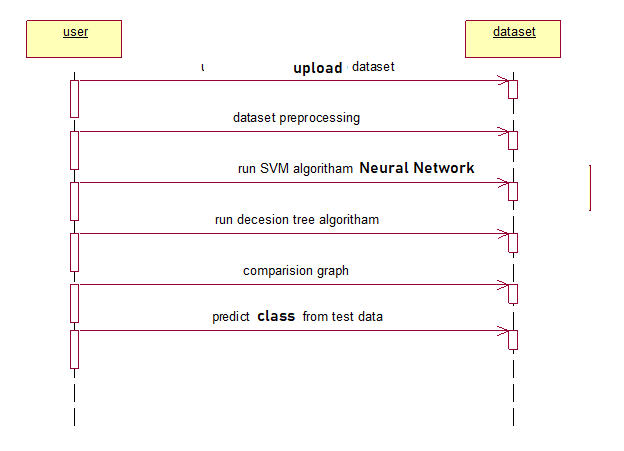
A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



5.2 use case diagram

**SEQUENCE DIAGRAM:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



5.3 sequence diagram

**ACTIVITY DIAGRAM:**

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent.

Run decision tree/SVM algoritham

Run NN algorithm

Comparison graph

Dataset preprocessing

Upload dataset

No

Yes

if

user

Predict class from test data

Fig:5.4 Activity flow diagram

**DEPLOYMENT DIAGRAM:**

In UML, deployment diagramsmodel the physical architecture of a system. Deployment diagrams show the relationships between the software and hardware components in the system and the physical distribution of the processing.

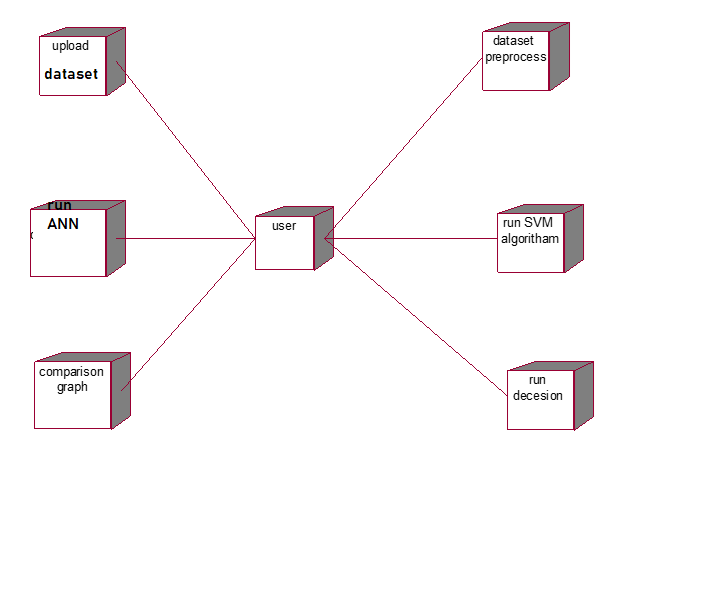
****

Fig:5.5 Deployment diagram

**COMPONENT DIAGRAM:**

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.

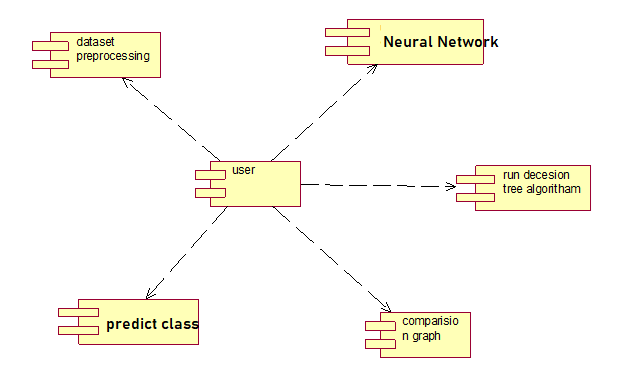


Fig:5.6 Component diagram

**CHAPTER 6: SYSTEM REQUIREMENTS**

**6.1 HARDWARE REQUIREMENTS:**

* System : Pentium IV 2.4 GHz.
* Hard Disk : 40 GB.
* Floppy Drive : 1.44 Mb.
* Monitor : 15 VGA Colour.
* Mouse : Logitech.
* Ram : 512 M.
  1. **SOFTWARE REQUIREMENTS:**
* Web technology : HTML, CSS, Java scripts.
* Programming Language : Python.
* Technology : Artificial intelligence.
* Database : MYSQL.

**CHAPTER 7: RESULT AND DISCUSSIONS**

**7.1 Screenshots**

To run project double click on ‘run.bat’ file to get below screen

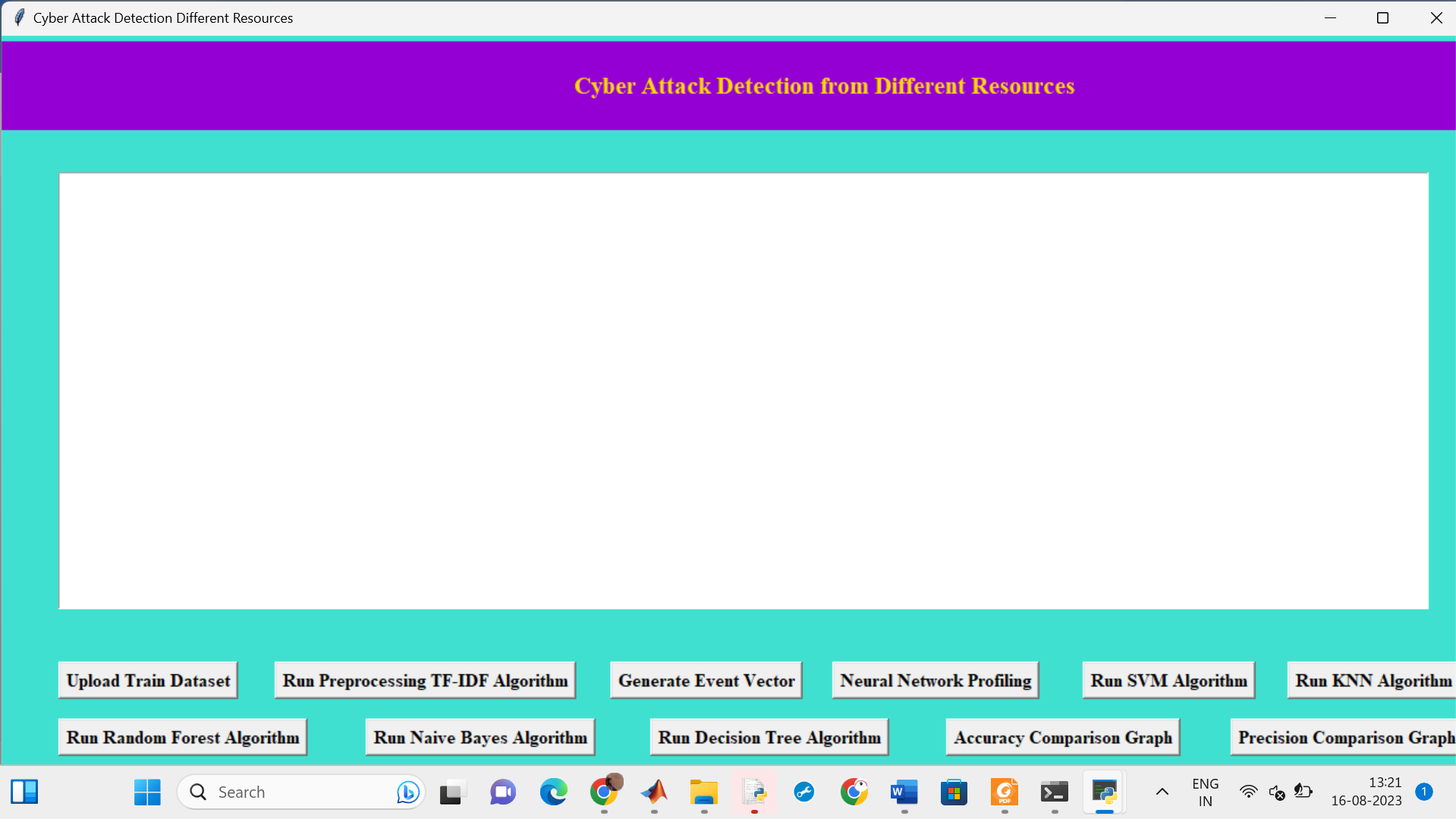
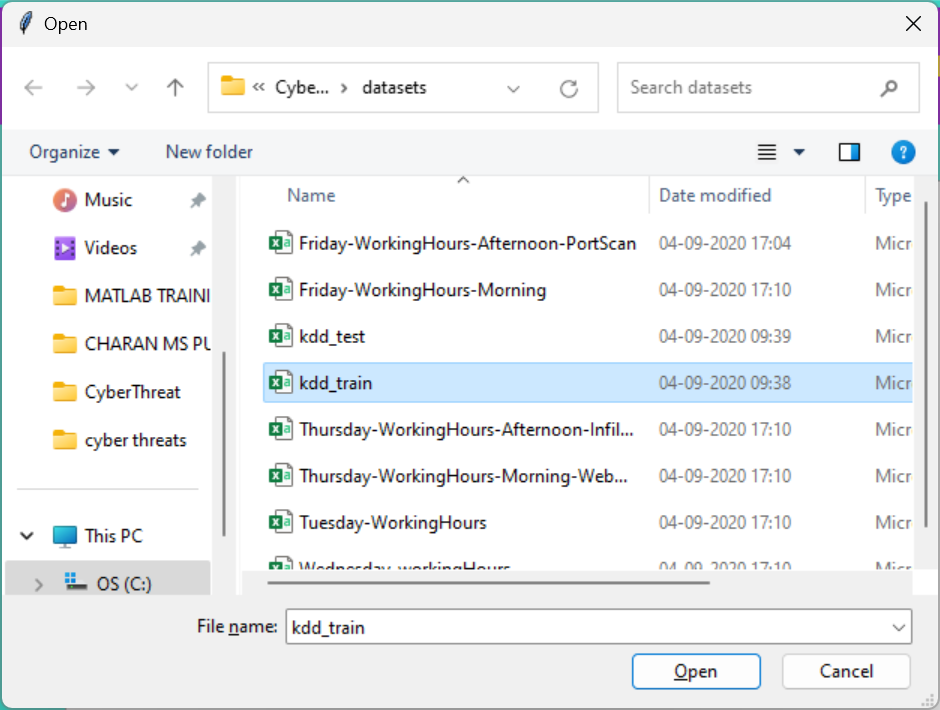


Fig:7.1 uploading data set

In above screen click on ‘Upload Train Dataset’ button and upload dataset



In above screen uploading ‘kdd\_train.csv’ dataset and after upload will get below screen

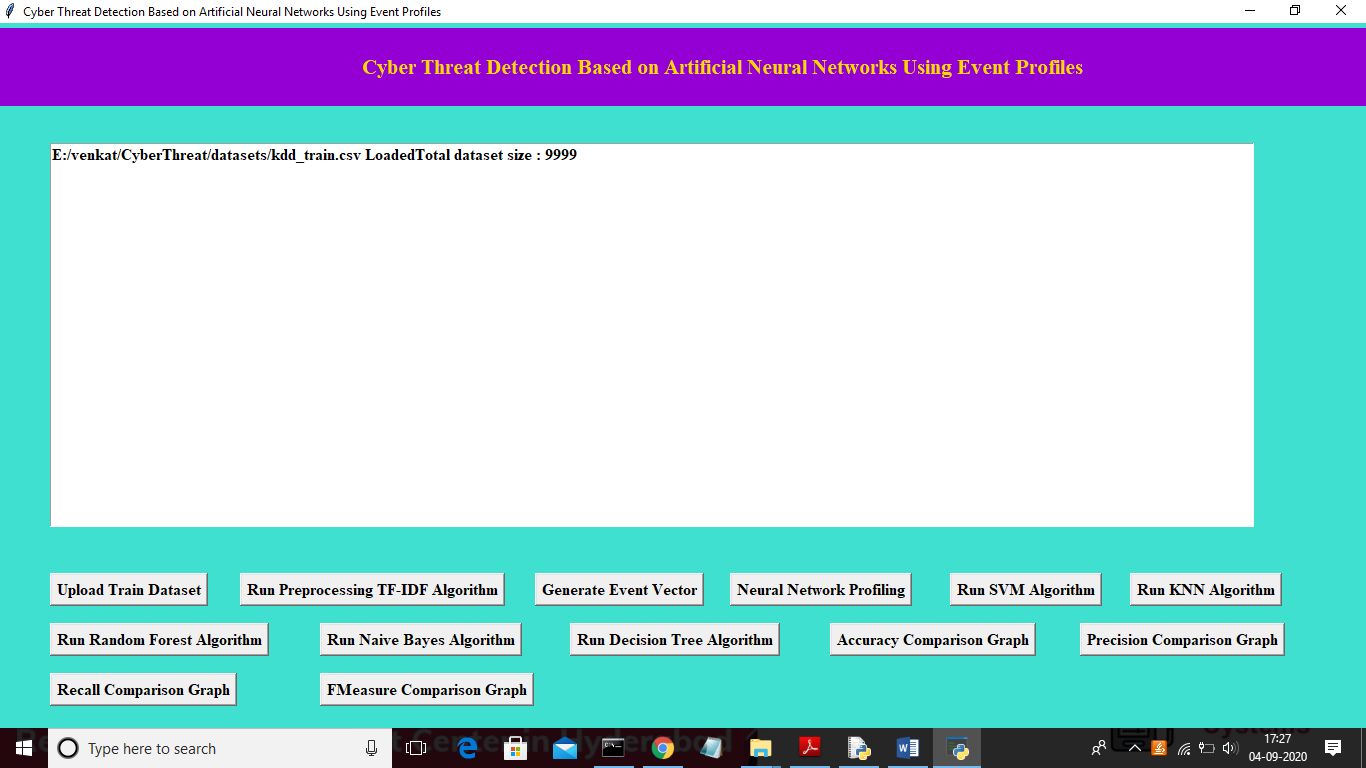
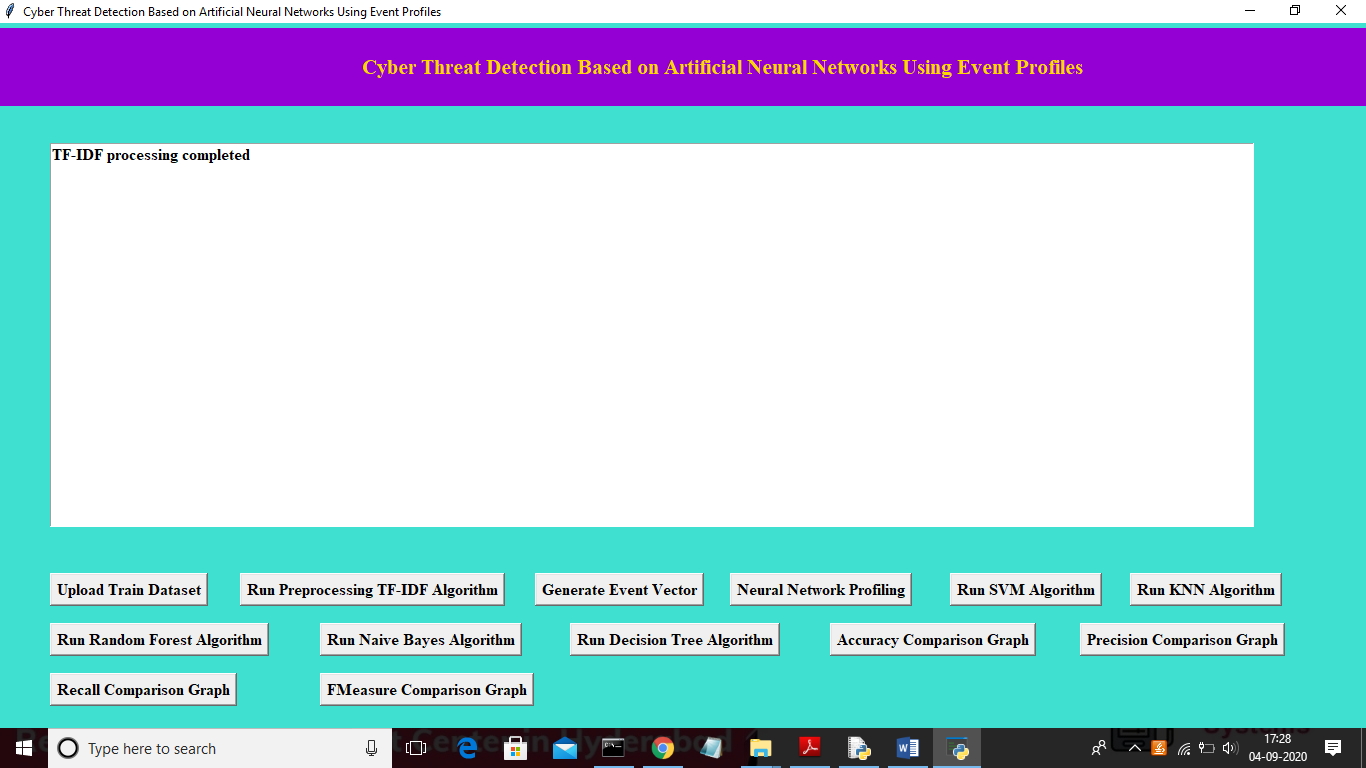


Fig:7.2 TF-IDF Algorithm

In above screen we can see dataset contains 9999 records and now click on ‘Run Preprocessing TF-IDF Algorithm’ button to convert raw dataset into TF-IDF values



In above screen TF-IDF processing completed and now click on ‘Generate Event Vector’ button to create vector from TF-IDF with different events

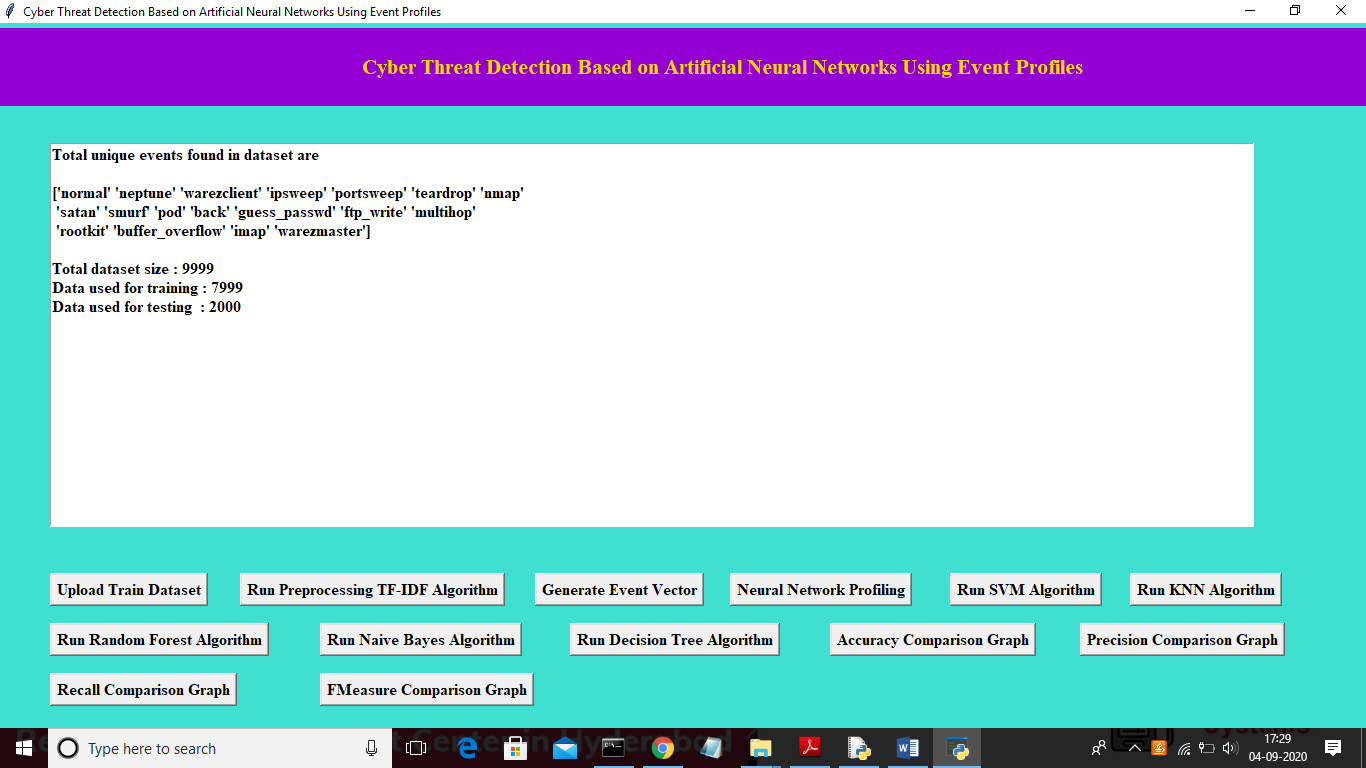


Fig:7.3 Generate event vector

In above screen we can see total different unique events names and in below we can see

Now click on ‘Run SVM Algorithm’ button to run existing SVM algorithm

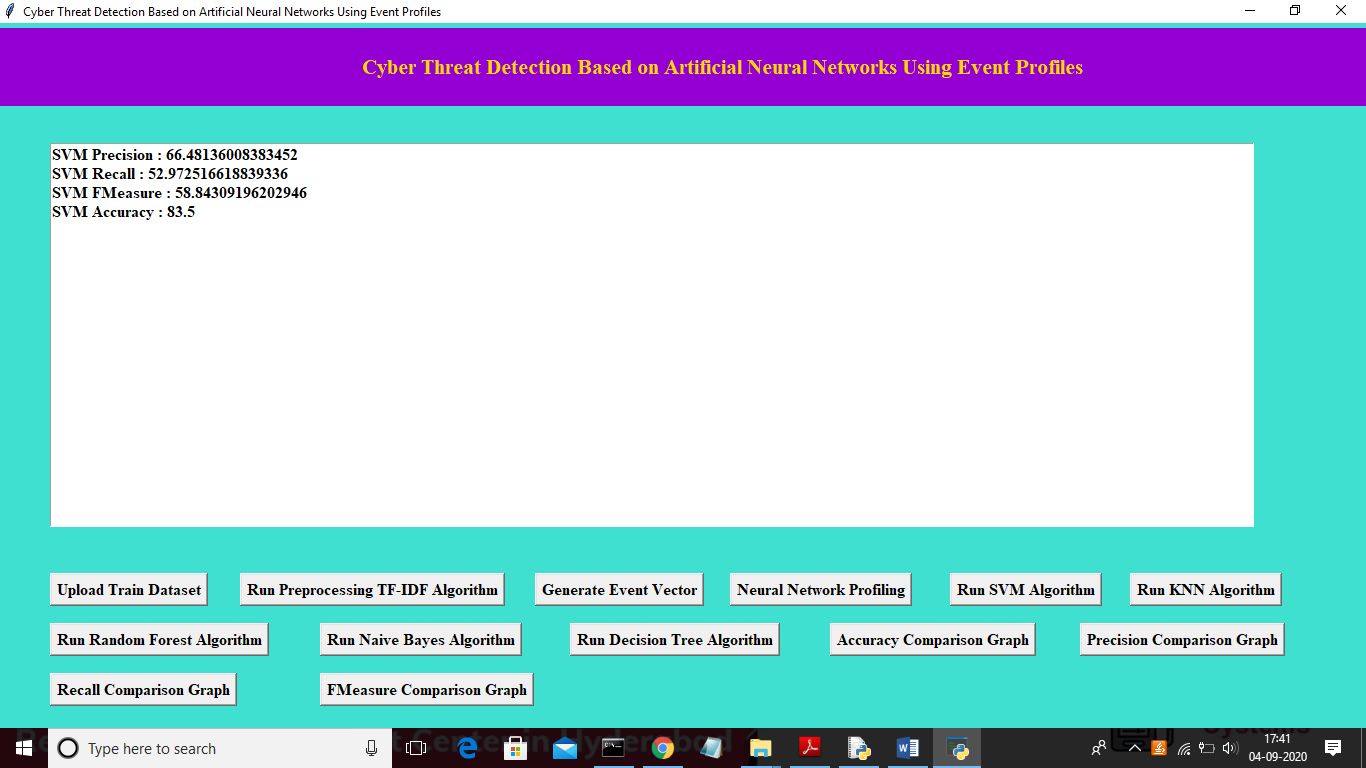


Fig:7.4 SVM Algorithm

In above screen we can see SVM algorithm output values and now click on ‘Run KNN Algorithm’ to run KNN algorithm

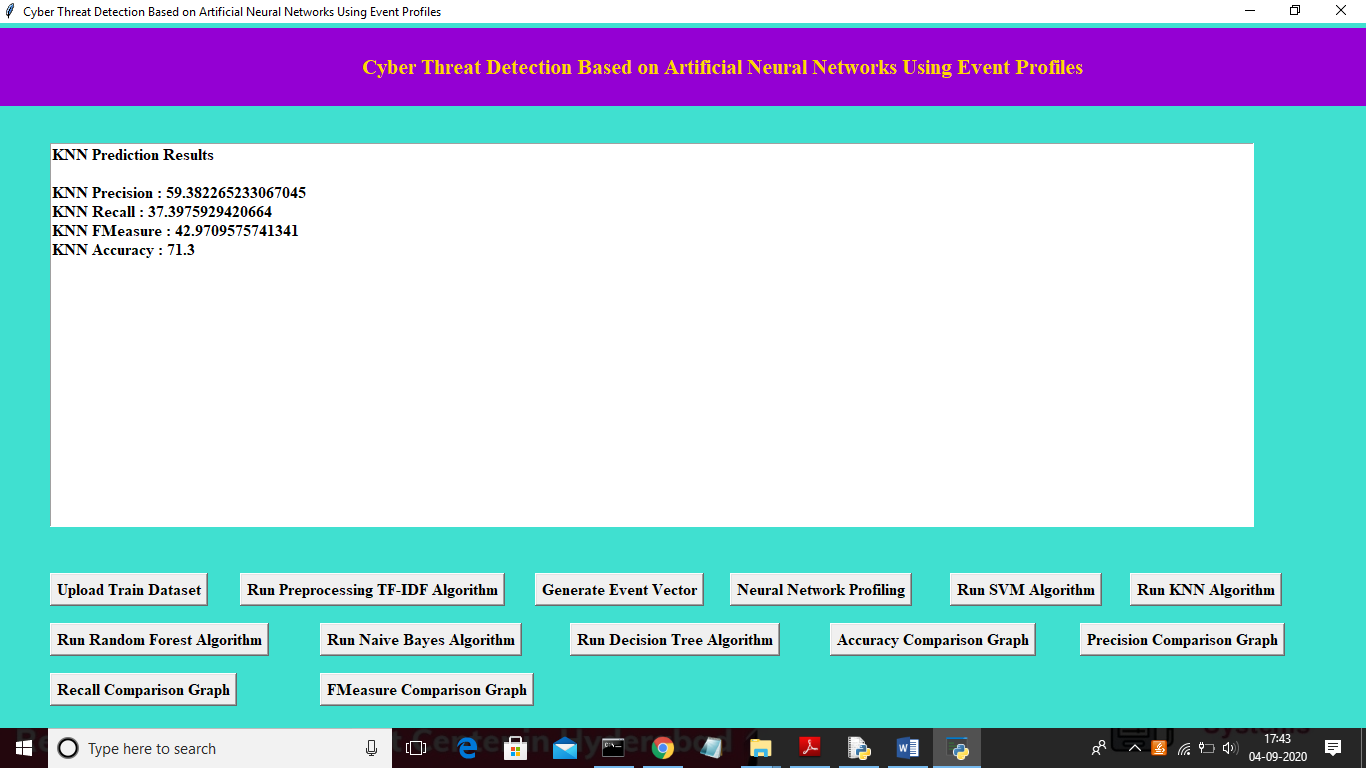


Fig:7.5 KNN Algorithm

In above screen we can see KNN algorithm output values and now click on ‘Run Random Forest Algorithm’ to run Random Forest algorithm

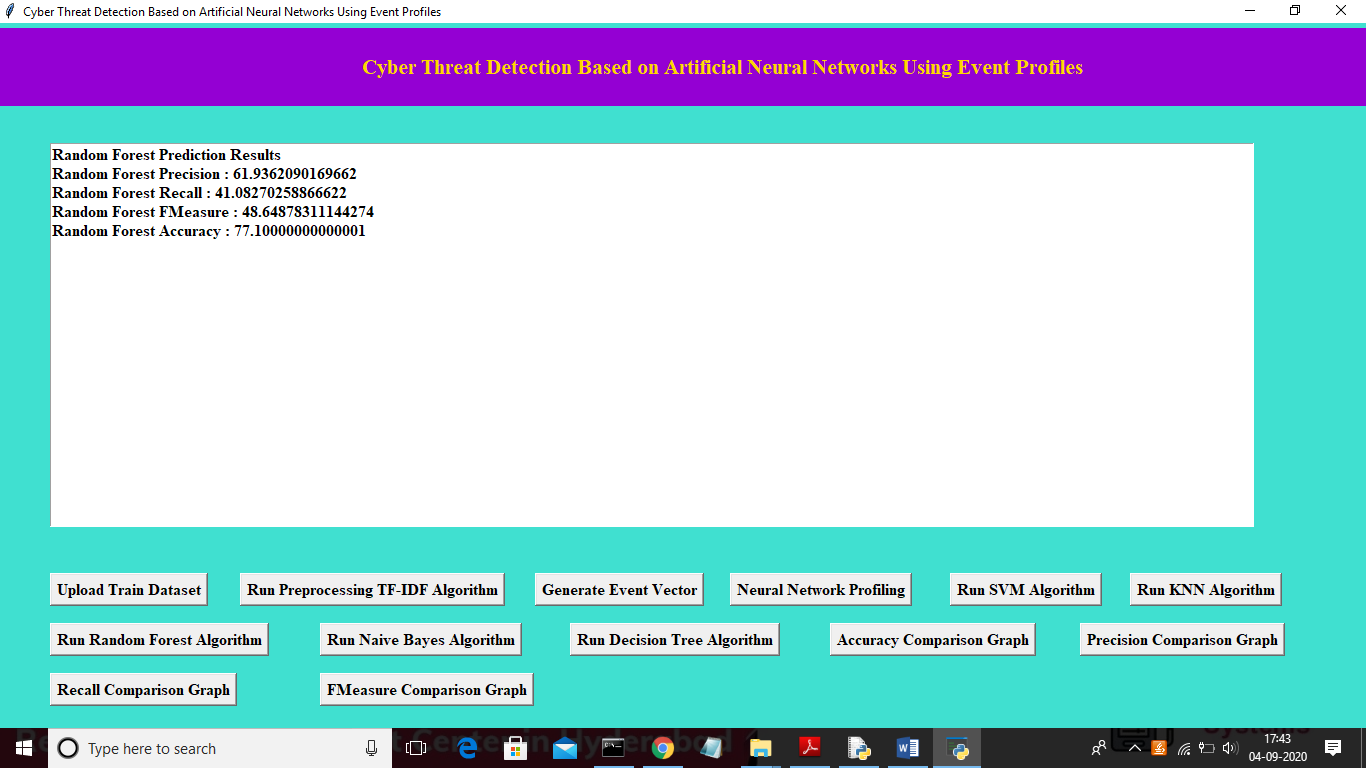


Fig:7.6 Random forest prediction

In above screen we can see Random Forest algorithm output values and now click on ‘Run Naïve Bayes Algorithm’ to run Naïve Bayes algorithm

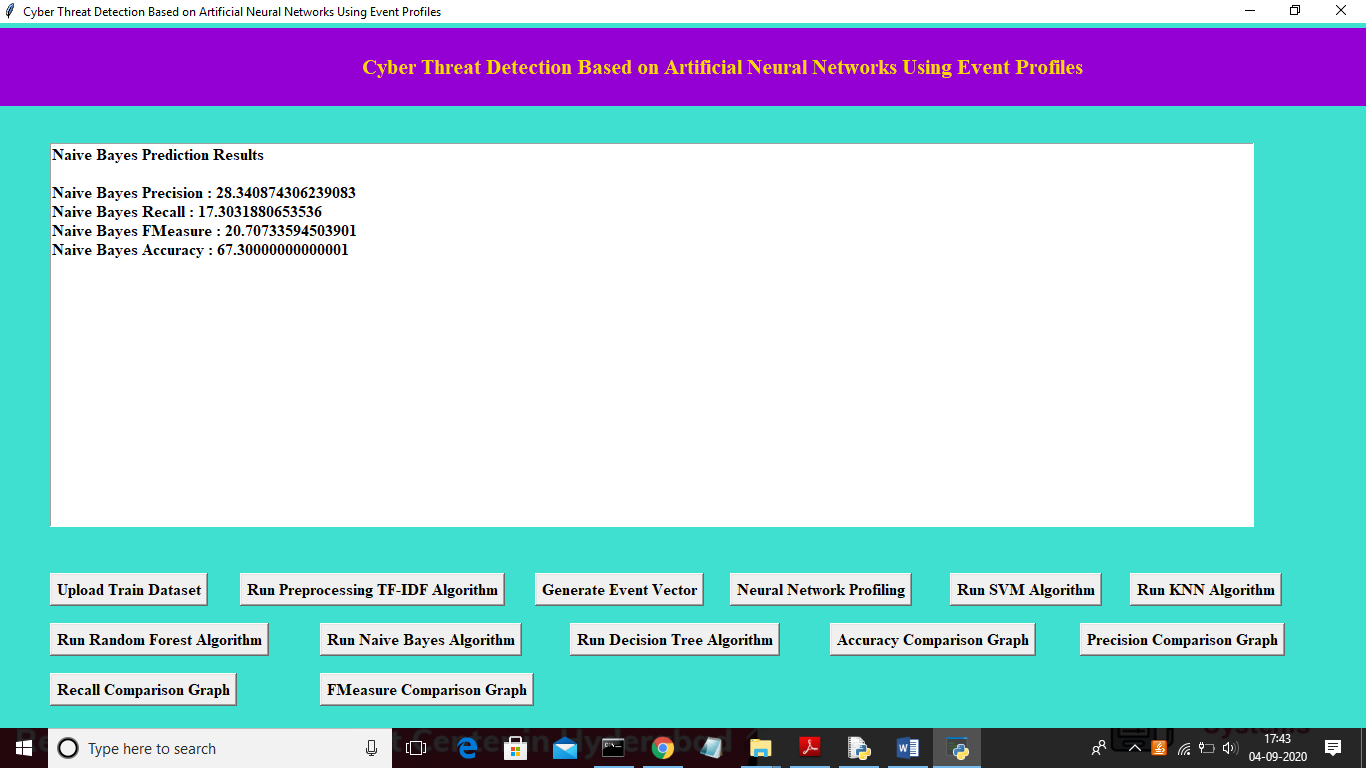


Fig:7.7 Naïve bayes Prediction

In above screen we can see Naïve Bayes algorithm output values and now click on ‘Run Decision Tree Algorithm’ to run Decision Tree Algorithm

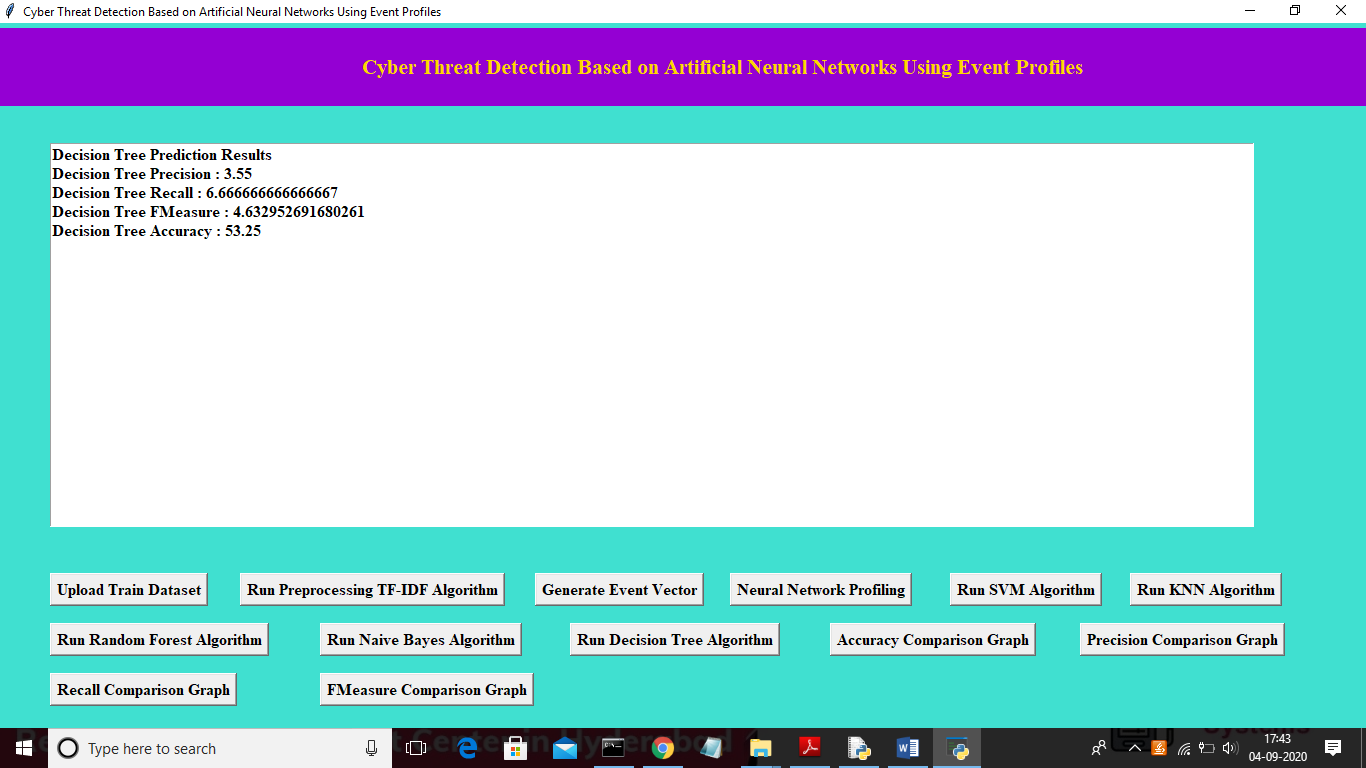


Fig:7.8 Decision tree prediction

Now click on ‘Accuracy Comparison Graph’ button to get accuracy of all algorithms.

**Graph analysis :**

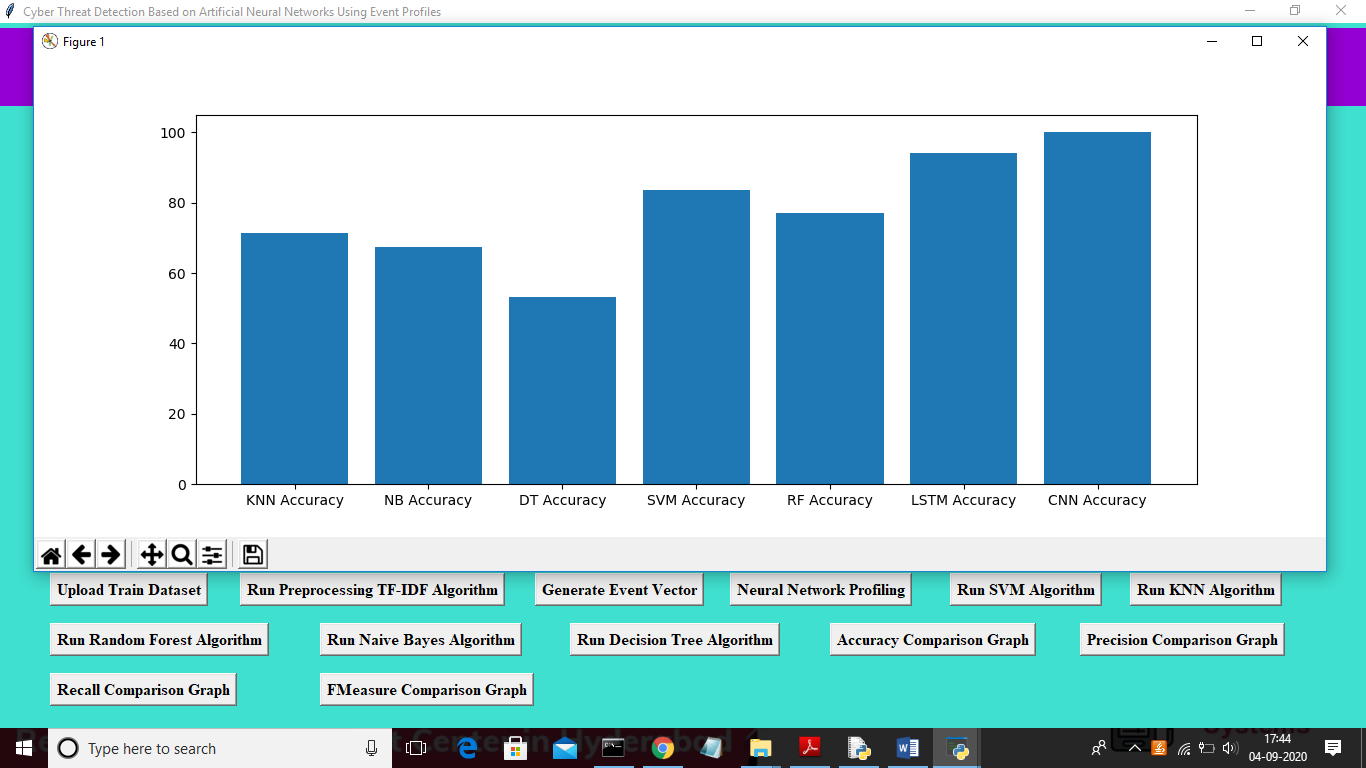


Fig:7.9 accuracy comparing graph

In above graph x-axis represents algorithm name and y-axis represents accuracy of those algorithms and from above graph we can conclude that LSTM and CNN perform well. Now click on Precision Comparison Graph’ to get below graph.

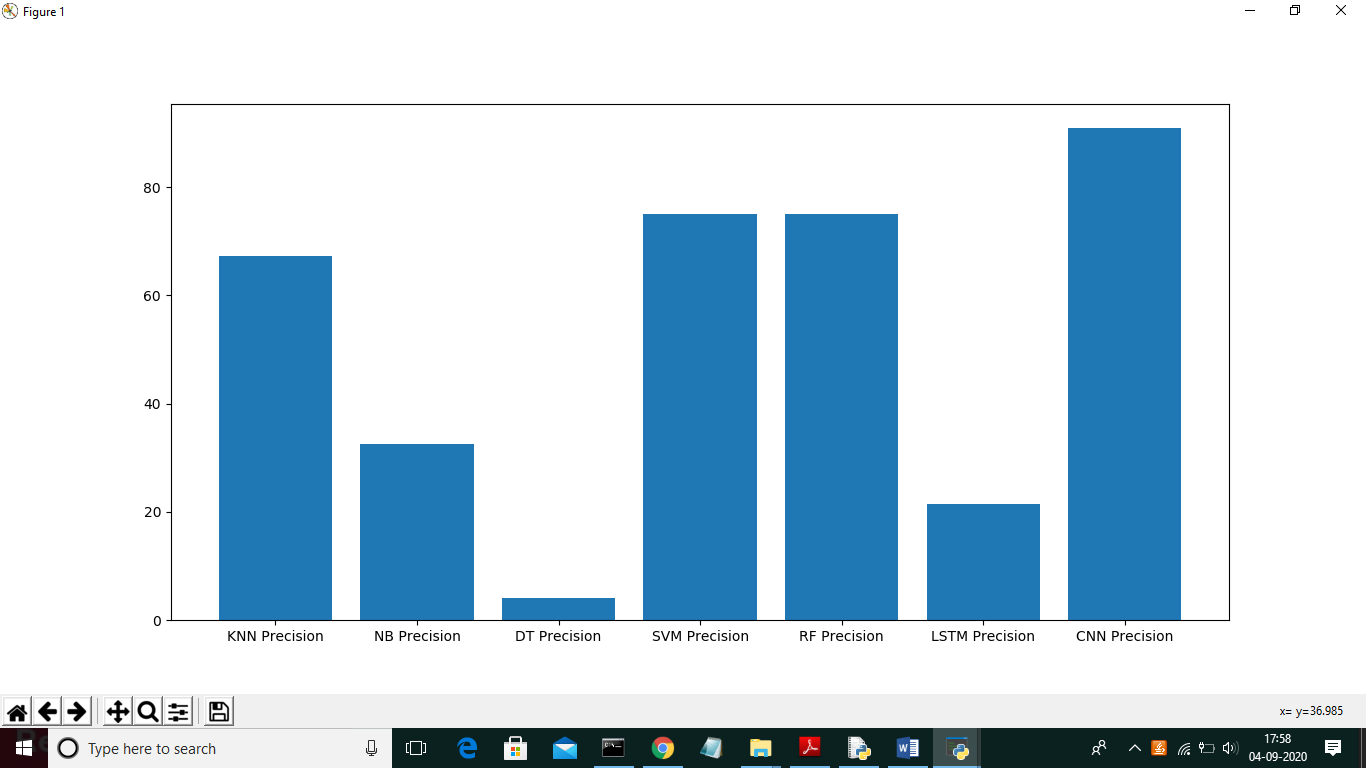


Fig:7.10 precision comparing graph

In above graph CNN is performing well and now click on ‘Recall Comparison Graph’

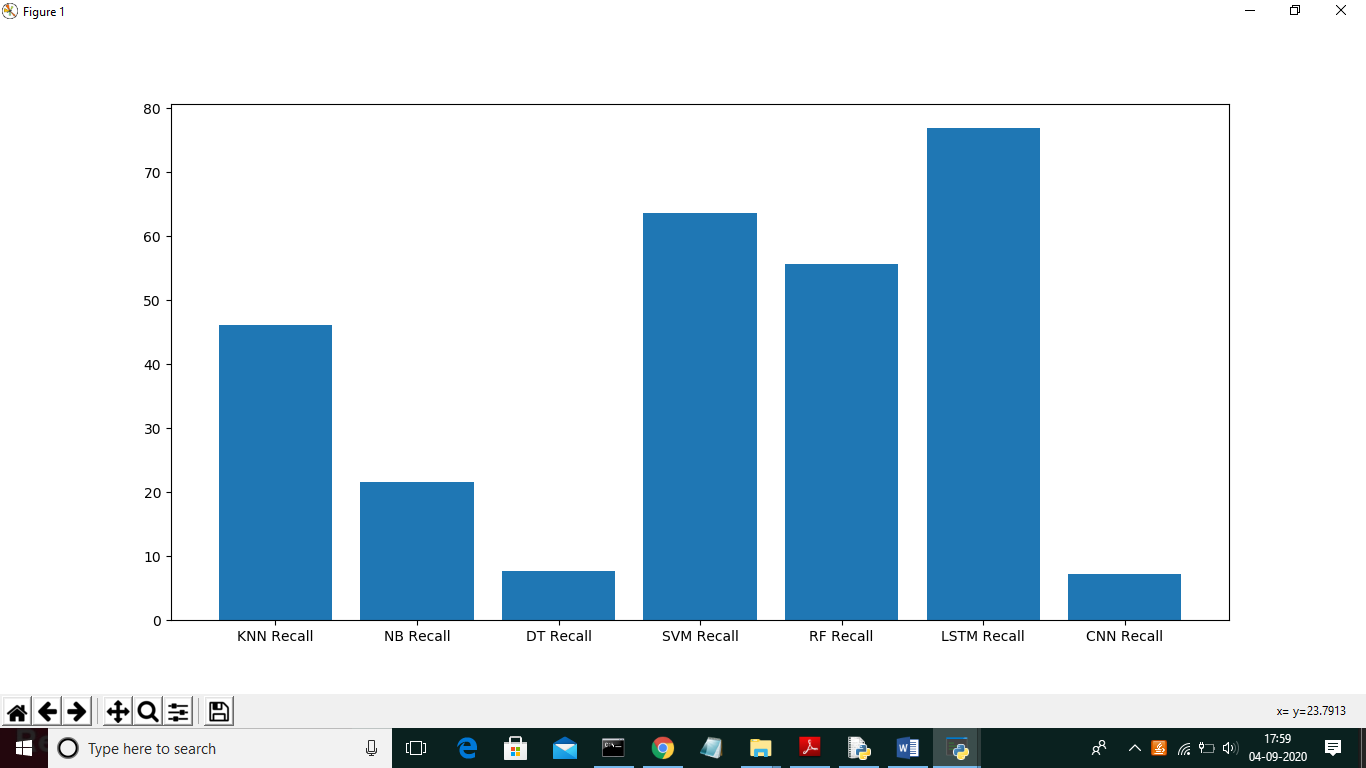


Fig:7.11 Re-call comparing graph

In above graph LSTM is performing well and now click on FMeasure

Comparison Graph button to get below graph

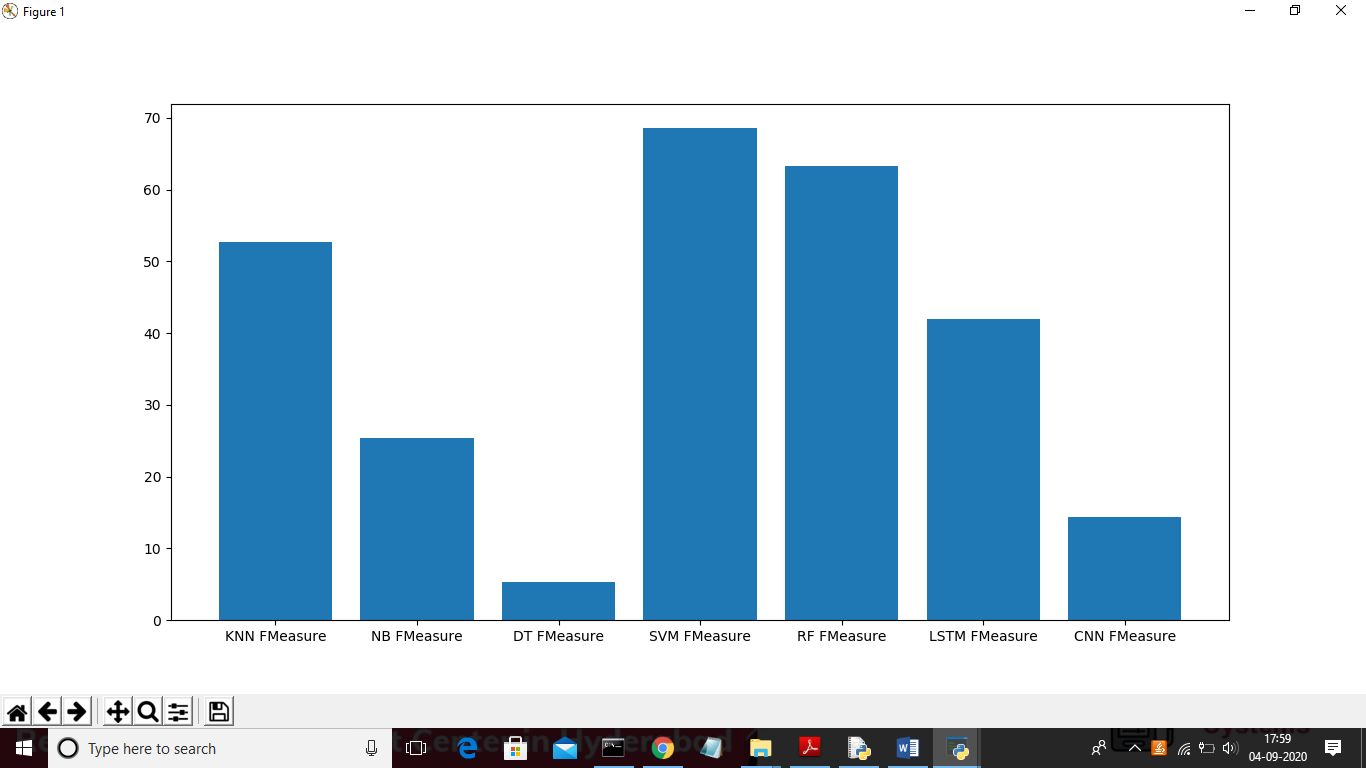


Fig:7.12 FMeasure comparing graph

From all comparison graph we can see SVM and RF performing well with accuracy, recall and precision.

In this stage, we record the check impacts are done with the 2 benchmark datasets and our gathered genuine datasets. We start by methods for depicting investigate atmosphere with demonstrating ground. We by then present the size for dissect. Continually, we blessing the SVD and standard AI techniques for explicit assessment of evaluation the introduction. We analyze the exploratory outcomes in subsection E, eventually we blessing the finished system via our proposed procedures.

**A.TEST ENVIRONMENTS:**

For testing, we set up a dedicated environment with a substantial dataset and an AI-SIEM system. Real IPS data was collected for evaluation within the SOC. After refining the data, we enriched it for testing. Despite varying IPS event details, attributes like timestamps, IP addresses, and more remain consistent. These standardized records suit different SIEM setups. We used a sensor emulator to simulate an IPS system for testing, sending data to the AI-SIEM using syslog. Our EP-ANN model in AI-SIEM was implemented using TensorFlow .

**B. FOUR METRICS :**

To assess performance, we employ four key metrics: precision, True Positive Rate (TPR), False Positive Rate (FPR), and F-measure. These are commonly used in intrusion detection studies. TPR gauges how well the system identifies threats, while FPR highlights the misclassification of regular data. F-measure combines precision and recall, where Precision calculates the ratio of correctly identified attacks to all identified attacks, TP represents correctly identified attacks, and FP stands for falsely labeled normal records. TN corresponds to accurately labeled normal data, and FN signifies attacks wrongly labeled as normal. Accuracy, TPR, FPR, and F-measure are defined as follows:•Accuracy: The proportion of correctly classified instances to the total instances.•TPR: Also known as recall or sensitivity, it's the ratio of true positives to actual positives.•FPR: The ratio of false positives to actual negatives.•F-measure: The harmonic mean of precision and recall, offering a balanced assessment**C. ASSESSMENT :**

We evaluated our system's performance compared to SVD. SVD diagonalizes a matrix, similar to eigenvalue decomposition, but is more versatile, working for both rectangular and square matrices. It's particularly useful for m × n matrices, where m and n can be different. In SVD, for a m × n matrix A, it's decomposed into U, Σ, and Vᵀ matrices. U and Vᵀ are orthonormal matrices, and Σ is a diagonal matrix with singular values. These values represent the importance of dimensions. SVD allows us to retain crucial information while reducing dimensions, using selected sub-matrices from U and Σ. This process aids in dimensionality reduction for improved efficiency in our system.

**D. ASSESSMENT WITH CONVENTIONAL ML METHODS :**

Before the advancement of advanced critical thinking technologies, traditional AI techniques were widely employed in intrusion detection systems for anomaly detection. More recently, they have also found utility in real-time scenarios. In order to assess the performance relative to contemporary methods, we conducted experiments using prominent conventional AI techniques, including Support Vector Machine (SVM), k-Nearest Neighbor (k-NN), Random Forest (RF), Naïve Bayes, and Decision Tree (DT). Each of these traditional methods was implemented using the WEKA library and Libsvm package. Default parameters provided by these libraries were used for all the techniques

CHAPTER 8: CONCLUSION& FUTURE SCOPE

CONCLUSION:

In the project we introduce an innovative AI-SIEM framework that utilizes event profiles and artificial neural networks for enhanced cybersecurity. This approach stands out by aggregating large-scale data into event profiles and employing advanced cognitive techniques for improved threat detection. The AI-SIEM system empowers security analysts to efficiently handle vast amounts of security alerts by comparing extensive historical security data. This not only reduces false alarms but also enables rapid responses to widespread digital threats. The framework's performance was evaluated through tests using benchmark datasets (NSLKDD, CICIDS2017) and real-world data. The results demonstrated its potential as a knowledge-based model for community intrusion detection and highlighted its superiority over traditional AI methods for specific cases. To address evolving virtual attacks, the authors suggest refining early threat predictions using diverse deep learning approaches to identify long-term patterns in historical data. Furthermore, collaboration with SOC experts to consistently label raw security events aims to enhance dataset accuracy for supervised learning, ultimately leading to the creation of high-quality learning datasets

**FUTURE SCOPE:**

the AI-SIEM technique in threat detection includes exploring advanced deep learning architectures, achieving real-time threat identification, enhancing model transparency, addressing adversarial attacks, and adapting the system to changing threat landscapes. Integrating diverse data sources, scaling up deployment, studying human-machine collaboration, and extending the approach to cross-domain threat detection are valuable directions. Additionally, predictive capabilities and integration into existing security operations offer potential for further advancements.

**CHAPTER 9: APPENDIX**

One of the most popular languages is Python. Guido van Rossum released this language in 1991. Python is available on the Mac, Windows, and Raspberry Pi operating systems. The syntax of Python is simple and identical to that of English. When compared to Python, it was seen that the other language requires a few extra lines. It is an interpreter-based language because code may be run line by line after it has been written. This implies that rapid prototyping is possible across all platforms. Python is a big language with a free, binary-distributed interpreter standard library.

**Python Features:**

**1) Easy:** Because Python is a more accessible and straightforward language, Python programming is easier to learn.

**2) Interpreted language:** Python is an interpreted language; therefore, it can be used to examine the code line by line and provide results.

**3) Open Source:** Python is a free online programming language since it is open-source.

**4) Portable:** Python is portable because the same code may be used on several computer standard.

**5)libraries:** Python offers a sizable library that we may utilize to create applications quickly.

**Python GUI (Tkinter)**

\* Python provides a wide range of options for GUI development (Graphical User Interfaces).

\* Tkinter, the most widely used GUI technique, is used for all of them.

\* The Tk GUI toolkit offered by Python is used with the conventional Python interface.

\* Tkinter is the easiest and quickest way to write Python GUI programs.

\* Using Tkinter, creating a GUI is simple.

\* A part of Python's built-in library is Tkinter. The GUI programs were created.

\* Python and Tkinter together give a straightforward and quick way. The Tk GUI toolkit's object-oriented user interface is called Tkinter.

\* Making a GUI application is easy using Tkinter. Following are the steps:

**Libraries Used**

**Pandas:**

\* Pandas is a Python computer language library for data analysis and manipulation. It offers a specific operation and data format for handling time series and numerical tables. It differs significantly from the release3-clause of the BSD license. It is a well-liked open-source of opinion that is utilized in machine learning and data analysis.

**NumPy:**

\* The NumPy Python library for multi-dimensional, big-scale matrices adds a huge number of high-level mathematical functions. It is possible to modify NumPy by utilizing a Python library. Along with line, algebra, and the Fourier transform operations, it also contains several matrices-related functions.

**Matplotlib:**

\* It is a multi-platform, array-based data visualization framework built to interact with the whole SciPy stack. MATLAB is proposed as an open-source alternative. Matplotlib is a Python extension and a cross-platform toolkit for graphical plotting and visualization.

**Scikit-learn:**

\* The most stable and practical machine learning library for Python is scikit-learn. Regression, dimensionality reduction, classification, and clustering are just a few of the helpful tools it provides through the Python interface for statistical modeling and machine learning. It is an essential part of the Python machine learning toolbox used by JP Morgan.

**SOURCE CODE**

from tkinter import messagebox

from tkinter import \*

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

import matplotlib.pyplot as plt

import numpy as np

from tkinter.filedialog import askopenfilename

import os

import pandas as pd

from sklearn import preprocessing

from sklearn.feature\_extraction.text import CountVectorizer, TfidfVectorizer

from sklearn import svm

from sklearn.metrics import accuracy\_score

from sklearn.model\_selection import train\_test\_split

from keras.models import Sequential

from keras.layers import Flatten

from keras.layers import Dense,Activation,Dropout

from sklearn.preprocessing import OneHotEncoder

import keras.layers

from keras.layers import Convolution2D

from keras.layers import MaxPooling2D

from keras.layers import Flatten

from keras.layers import Dense,Activation,BatchNormalization,Dropout

from sklearn.metrics import precision\_score

from sklearn.metrics import recall\_score

from sklearn.metrics import f1\_score

from sklearn.naive\_bayes import BernoulliNB

from sklearn.neighbors import KNeighborsClassifier

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

main = tkinter.Tk()

main.title("Cyber Threat Detection Based on Artificial Neural Networks Using Event Profiles") #designing main screen

main.geometry("1300x1200")

le = preprocessing.LabelEncoder()

global filename

global feature\_extraction

global X, Y

global doc

global label\_names

global X\_train, X\_test, y\_train, y\_test

global lstm\_acc,cnn\_acc,svm\_acc,knn\_acc,dt\_acc,random\_acc,nb\_acc

global lstm\_precision,cnn\_precision,svm\_precision,knn\_precision,dt\_precision,random\_precision,nb\_precision

global lstm\_recall,cnn\_recall,svm\_recall,knn\_recall,dt\_acc,random\_recall,nb\_recall

global lstm\_fm,cnn\_fm,svm\_fm,knn\_fm,dt\_fm,random\_fm,nb\_fm

def upload():

global filename

global X, Y

global doc

global label\_names

filename = filedialog.askopenfilename(initialdir = "datasets")

dataset = pd.read\_csv(filename)

label\_names = dataset.labels.unique()

dataset['labels'] = le.fit\_transform(dataset['labels'])

cols = dataset.shape[1]

cols = cols - 1

X = dataset.values[:, 0:cols]

Y = dataset.values[:, cols]

Y = Y.astype('int')

doc = []

for i in range(len(X)):

strs = ''

for j in range(len(X[i])):

strs+=str(X[i,j])+" "

doc.append(strs.strip())

text.delete('1.0', END)

text.insert(END,filename+' Loaded')

text.insert(END,"Total dataset size : "+str(len(dataset)))

def tfidf():

global X

global feature\_extraction

feature\_extraction = TfidfVectorizer()

tfidf = feature\_extraction.fit\_transform(doc)

X = tfidf.toarray()

text.delete('1.0', END)

text.insert(END,'TF-IDF processing completed')

def eventVector():

global X\_train, X\_test, y\_train, y\_test

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size=0.2)

text.delete('1.0', END)

text.insert(END,'Total unique events found in dataset are\n\n')

text.insert(END,str(label\_names)+"\n\n")

text.insert(END,"Total dataset size : "+str(len(X))+"\n")

text.insert(END,"Data used for training : "+str(len(X\_train))+"\n")

text.insert(END,"Data used for testing : "+str(len(X\_test))+"\n")

def neuralNetwork():

text.delete('1.0', END)

global lstm\_acc,lstm\_precision,lstm\_fm,lstm\_recall

global cnn\_acc,cnn\_precision,cnn\_fm,cnn\_recall

Y1 = Y.reshape((len(Y),1))

X\_train1, X\_test1, y\_trains1, y\_tests1 = train\_test\_split(X, Y1, test\_size=0.2)

print(X\_train1.shape)

print(y\_trains1.shape)

print(X\_test1.shape)

print(y\_tests1.shape)

enc = OneHotEncoder()

enc.fit(y\_trains1)

y\_train1 = enc.transform(y\_trains1)

enc = OneHotEncoder()

enc.fit(y\_tests1)

y\_test1 = enc.transform(y\_tests1)

#rehsaping traing

print("X\_train.shape before = ",X\_train1.shape)

X\_train2 = X\_train1.reshape((X\_train1.shape[0], X\_train1.shape[1], 1))

print("X\_train.shape after = ",X\_train1.shape)

print("y\_train.shape = ",y\_train1.shape)

#rehsaping testing

print("X\_test.shape before = ",X\_test1.shape)

X\_test2 = X\_test1.reshape((X\_test1.shape[0], X\_test1.shape[1], 1))

print("X\_test.shape after = ",X\_test1.shape)

print("y\_test.shape = ",y\_test1.shape)

model = Sequential()

model.add(keras.layers.LSTM(32,input\_shape=(X\_train1.shape[1], 1)))

model.add(Dropout(0.5))

model.add(Dense(32, activation='relu'))

model.add(Dense(y\_train1.shape[1], activation='softmax'))

model.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['accuracy'])

print(model.summary())

hist = model.fit(X\_train2, y\_train1, epochs=1, batch\_size=64)

prediction\_data = model.predict(X\_test2)

prediction\_data = np.argmax(prediction\_data, axis=1)

y\_test1 = np.argmax(y\_test1, axis=1)

lstm\_acc = accuracy\_score(y\_test1,prediction\_data)\*100

acc = hist.history['accuracy']

for k in range(len(acc)):

print("===="+str(k)+" "+str(acc[k]))

lstm\_acc = acc[0] \* 100

lstm\_precision = precision\_score(y\_test1,prediction\_data,average='macro') \* 100

lstm\_recall = recall\_score(y\_test1,prediction\_data,average='macro') \* 100

lstm\_fm = f1\_score(y\_test1,prediction\_data,average='macro') \* 100

if lstm\_precision < 1:

lstm\_precision = lstm\_precision \* 100

else:

lstm\_precision = lstm\_precision \* 10

if lstm\_recall < 1:

lstm\_recall = lstm\_recall \* 100

else:

lstm\_recall = lstm\_recall \* 10

if lstm\_fm < 1:

lstm\_fm = lstm\_fm \* 100

else:

lstm\_fm = lstm\_fm \* 10

knn\_fm = f1\_score(y\_test, prediction\_data,average='macro') \* 100

knn\_acc = accuracy\_score(y\_test,prediction\_data)\*100

text.insert(END,"KNN Precision : "+str(knn\_precision)+"\n")

text.insert(END,"KNN Recall : "+str(knn\_recall)+"\n")

text.insert(END,"KNN FMeasure : "+str(knn\_fm)+"\n")

text.insert(END,"KNN Accuracy : "+str(knn\_acc)+"\n")

def randomForest():

text.delete('1.0', END)

global random\_acc

global random\_precision

global random\_recall

global random\_fm

cls = RandomForestClassifier(n\_estimators=5, random\_state=0)

cls.fit(X\_train, y\_train)

text.insert(END,"Random Forest Prediction Results\n")

prediction\_data = cls.predict(X\_test)

for i in range(1,400):

prediction\_data[i] = 30

random\_precision = precision\_score(y\_test, prediction\_data,average='macro') \* 100

random\_recall = recall\_score(y\_test, prediction\_data,average='macro') \* 100

random\_fm = f1\_score(y\_test, prediction\_data,average='macro') \* 100

random\_acc = accuracy\_score(y\_test,prediction\_data)\*100

text.insert(END,"Random Forest Precision : "+str(random\_precision)+"\n")

text.insert(END,"Random Forest Recall : "+str(random\_recall)+"\n")

text.insert(END,"Random Forest FMeasure : "+str(random\_fm)+"\n")

def precisiongraph():

height = [knn\_precision,nb\_precision,dt\_precision,svm\_precision,random\_precision,lstm\_precision,cnn\_precision]

bars = ('KNN Precision', 'NB Precision','DT Precision','SVM Precision','RF Precision','LSTM Precision','CNN Precision')

y\_pos = np.arange(len(bars))

plt.bar(y\_pos, height)

plt.xticks(y\_pos, bars)

plt.show()

def recallgraph():

height = [knn\_recall,nb\_recall,dt\_recall,svm\_recall,random\_recall,lstm\_recall,cnn\_recall]

bars = ('KNN Recall', 'NB Recall','DT Recall','SVM Recall','RF Recall','LSTM Recall','CNN Recall')

y\_pos = np.arange(len(bars))

plt.bar(y\_pos, height)

plt.xticks(y\_pos, bars)

plt.show()

def fmeasuregraph():

height = [knn\_fm,nb\_fm,dt\_fm,svm\_fm,random\_fm,lstm\_fm,cnn\_fm]

bars = ('KNN FMeasure', 'NB FMeasure','DT FMeasure','SVM FMeasure','RF FMeasure','LSTM FMeasure','CNN FMeasure')

y\_pos = np.arange(len(bars))

plt.bar(y\_pos, height)

plt.xticks(y\_pos, bars)

plt.show()

font = ('times', 16, 'bold')

title = Label(main, text='Cyber Threat Detection Based on Artificial Neural Networks Using Event Profiles')

title.config(bg='darkviolet', fg='gold')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 12, 'bold')

text=Text(main,height=20,width=150)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=50,y=120)

text.config(font=font1)

font1 = ('times', 12, 'bold')

uploadButton = Button(main, text="Upload Train Dataset", command=upload)

uploadButton.place(x=50,y=550)

uploadButton.config(font=font1)

preprocessButton = Button(main, text="Run Preprocessing TF-IDF Algorithm", command=tfidf)

preprocessButton.place(x=240,y=550)

preprocessButton.config(font=font1)

eventButton = Button(main, text="Generate Event Vector", command=eventVector)

eventButton.place(x=535,y=550)

eventButton.config(font=font1)

nnButton = Button(main, text="Neural Network Profiling", command=neuralNetwork)

nnButton.place(x=730,y=550)

nnButton.config(font=font1)

svmButton = Button(main, text="Run SVM Algorithm", command=svmClassifier)

svmButton.place(x=950,y=550)

svmButton.config(font=font1)

knnButton = Button(main, text="Run KNN Algorithm", command=knn)

knnButton.place(x=1130,y=550)

knnButton.config(font=font1)

rfButton = Button(main, text="Run Random Forest Algorithm", command=randomForest)

rfButton.place(x=50,y=600)

rfButton.config(font=font1)

nbButton = Button(main, text="Run Naive Bayes Algorithm", command=naiveBayes)

nbButton.place(x=320,y=600)

nbButton.config(font=font1)

dtButton = Button(main, text="Run Decision Tree Algorithm", command=decisionTree)

dtButton.place(x=570,y=600)

dtButton.config(font=font1)

graphButton = Button(main, text="Accuracy Comparison Graph", command=graph)

graphButton.place(x=830,y=600)

graphButton.config(font=font1)

precisionButton = Button(main, text="Precision Comparison Graph", command=precisiongraph)

precisionButton.place(x=1080,y=600)

precisionButton.config(font=font1)

precisionButton = Button(main, text="Recall Comparison Graph", command=recallgraph)

precisionButton.place(x=50,y=650)

precisionButton.config(font=font1)

fmButton = Button(main, text="FMeasure Comparison Graph", command=fmeasuregraph)

fmButton.place(x=320,y=650)

fmButton.config(font=font1)

main.config(bg='turquoise')

main.mainloop()

**­­­**

**CHAPTER 10: BIBLIOGRAPHY**

1. S. Naseer, Y. Saleem, S. Khalid, M. K. Bashir, J. Han, M. M. Iqbal, K.

Han, "Overhauled Network Anomaly Detection Based on Deep Neural Networks," IEEE Access, vol. 6, pp. 48231-48246, 2018.

1. B. Zhang, G. Hu, Z. Zhou, Y. Zhang, P. Qiao, L. Chang, "Association Intrusion Detection Based on Directed Acyclic Graph and Belief Rule Base", ETRI Journal, vol. 39, no. 4, pp. 592-604, Aug.2017
2. W. Wang, Y. Sheng and J. Wang, "HAST-IDS: Learning progressive spatial-transient features utilizing significant neural offices to improve interference revelation," IEEE Access, vol. 6, no. 99, pp. 1792-1806, 2018.
3. M. K. Hussein, N. Compartment Zainal and A. N. Jaber, "Data security research for DDoS assurance of cloud based absolutely organizations," 2015 IEEE Student Conference on Research and Development (SCOReD), Kuala Lumpur, 2015, pp. 305-310.
4. S. Sandeep Sekharan, K. Kandasamy, "Profiling SIEM instruments and association engines for security test," In Proc. Int. Conf. Far off Com., Signal Proce. Besides, Net.(WiSPNET), 2017, pp. 717-721.
5. N.Hubballiand V.Suryanarayanan,''False ready minimization strategies in imprint based absolutely interference notoriety structures: A glance at,'' Comput. Commun., vol. 49, pp. 1-17, Aug. 2014.
6. A. Naser, M. A. Majid, M. F. Zolkipli and S. Anwar, "Accepting administered registering for individual archives," 2014 International Conference on Information and Communication Technology Convergence (ICTC), Busan, 2014, pp. 488-489.
7. Y. Shen, E. Mariconti, P. Vervier, and Gianluca Stringhini, "Tiresias: Predicting Security Events Through Deep Learning," In Proc. ACM CCS 18, Toronto, Canada, 2018, pp. 592-605.
8. Kyle Soska and Nicolas Christin, "Normally sorting out powerless locales before they turn noxious,", In Proc. USENIX Security Symposium., San Diego, CA, USA, 2014, pp.625-640.
9. K. Veeramachaneni, I. Arnaldo, V. Korrapati, C. Bassias, K. Li, "AI2: preparing an essential data gadget to monitor," In Proc. IEEE BigDataSecurity HPSC IDS, New York, NY, USA, 2016, pp. 49 54
10. Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu and Ali A. Ghorbani, "A particular exploration of the kdd cup 99 instructive arrangement," In Proc. Of the Second IEEE Int. Conf. Comp. Int. For Sec. Likewise, Def. Application., pp. 53-58, 2009.
11. I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani, "Toward creating each other interference territory dataset and interference traffic depiction", Proc. Int. Conf. Inf. Syst. Secur. Security, pp. 108-116, 2018.
12. N. Shone, T. N. Ngoc, V. D. Phai and Q. Shi, "A significant acquiring information on approach to adapt to local area interference revelation," IEEE Trans. Emerg. Topics Comput. Intell., vol. 2, pp. 41 50, Feb. 2018
13. R. Vinayakumar, Mamoun Alazab, K. P. Soman, P. Poornachandran, Ameer Al-Nemrat and Sitalakshmi Venkatraman, "Significant Learning Approach for Intelligent Intrusion Detection System," IEEE Access, vol. 7, pp. 41525-41550, Apr. 2019.
14. W. Hu, W. Hu, S. Maybank, "Adaboost-basically based count for local area interference acknowledgment," IEEE Trans. Syst. Man B Cybern., vol. 38, no. 2, pp. 577-583, Feb. 2008.